

Create Rubric

92 points

List View

Grid View

i Create your rubric now or come back to it later. You can also make edits to your rubric while grading.

Q1 Honor Code and Preliminaries

0 points

Directions and Notes:

During this exam you will have 3 hours. Note that there won't be TA support available, so do your best if you're not sure about a question.

Along with this exam, you may reference your *handwritten* cheatsheets, but no other outside materials. (A digital version of handwritten notes would be OK.)

You may have a digital version of the CS88 Reference Sheet and your handwritten notes, but no other files.

Reminder: Please turn on your camera and share you *entire* screen. Do not share only your browser window.

Good luck!

As a member of the UC Berkeley community, I act with honesty, integrity, and respect for others.

— *UC Berkeley Honor Code*

By typing my name below, I am affirming that all work in this final is my own. I have not (and will not) use any resources outside of the allowed

1 +3.0
Thank you!

2 +0.0
Correct

3 +0.0
Incorrect

+ Add Rubric Item

materials, nor will I collaborate with anyone else during the exam.

Q2 Conceptual Questions

20 points

Select the correct answer for each of these questions.

Q2.1

2 points


True or False: A *generator* allows you to represent a finite or infinite stream of data.

True

False

Explanation

True. A generator which never raises a `StopIteration`, or a function that yields in a while loop can continue forever.

 Removing the **Correct** answer with auto-grading for this question.

1 +2.0
Correct


2 +0.0
Incorrect

 Add Rubric Item

Q2.2

2 points

What Python *keyword* turns a function into a generator?

 Removing the **Correct** answer with auto-grading for this question.

next

return

generate

yield

Explanation

yield is how a generator passes a result to its caller.

1 +2.0
Correct

2 +0.0
Incorrect

[+ Add Rubric Item](#)

Q2.3

2 points


True or False: The reduce function in Python *can* return a sequence.

True

False

Explanation

True. For example, you can reduce using a function like `append` over a list of lists. A list is a specific type of sequence. Though, perhaps a bit more pedantic as less interesting, a *string* is also a sequence! We can iterate over and index into a string, just like a list.

 Removing the **Correct** answer with auto-grading for this question.

1 +2.0
Correct


2 +0.0
Incorrect

[+ Add Rubric Item](#)

Q2.4

2 points

True or False: The filter function in Python *always* returns a sequence.

 Removing the **Correct** answer with auto-grading for this question.

True

False

1 +2.0
Correct

Explanation

True. The filter function returns a sequence of 0 or more items.

2 +0.0
Incorrect

[+ Add Rubric Item](#)


Q2.5

2 points

True or False: When an *exception* is raised by some function, our program *always* stops executing.

True

False

 Removing the **Correct** answer with auto-grading for this item.

1 +2.0
Correct

Explanation

False. Using `try / except` we can "catch" an error and continue execution. If we don't catch the error, our program will stop.


2 +0.0
Incorrect

[+ Add Rubric Item](#)

Q2.6

2 points

In the "Maps" project, you primarily practiced which programming paradigm?

 Removing the **Correct** answer with auto-grading for this item.

1 +2.0
Correct

Imperative Programming

Array-Based Programming

Functional Programming

Object-Oriented Programming

Declarative Programming

2 +0.0

Incorrect

[+ Add Rubric Item](#)

Explanation

The Maps project is primarily functional programming - notable the use of abstract data types, and patterns like using map, filter, zip, and similar functions.

Q2.7

2 points

In the "Ants" project, you primarily practiced which programming paradigm?

Imperative Programming

Array-Based Programming

Functional Programming

Object-Oriented Programming

Declarative Programming

⚠ Removing the **Correct** answer will remove this item from auto-grading for this assignment.

1 +2.0

Correct

2 +0.0

Incorrect

[+ Add Rubric Item](#)

Explanation

Ants is primarily a OOP-based project.

Q2.8

2 points

Of these items, which1 does every recursive function need?

- return statement
- yield statement
- if statement
- base case

Explanation

Every recursive function needs a *base case*, when a function stops making recursive calls. Most functions do use if statements and return a value, but those are technically not necessary.

⚠ Removing the **Correct** answer with auto-grading for this question

1 +2.0
Correct

2 +0.0
Incorrect

+ Add Rubric Item

Q2.9

2 points

Given a color Abstract Data Type, answer some questions based on this definition:

```
def color(r, g, b):
    return (r, g, b)
```

```
r = lambda color: color[0]
g = lambda color: color[1]
b = lambda color: color[2]
```

Does the g function violate the abstraction barrier?

Yes

No

⚠ Removing the **Correct** answer with auto-grading for this question

1 +2.0
Correct

2 +0.0
Incorrect

+ Add Rubric Item

Explanation

No, because these are "selectors" for our data. They are intrinsically linked to the structure of our data.

Q2.10

2 points

Consider a new function `combine_colors` with the following definition:

```
def combine_colors(color1, color2):
    return color(
        r(color1) + r(color2),
        g(color1) + g(color2),
        b(color1) + b(color2)
    )
```

Does the `combine_colors` function violate the abstraction barrier?

Yes

No

Explanation

Nope. This function sticks to using on the provided selectors, and assuming `r()`, `g()`, and `b()` are aligned with the `color()` function, then our `combine_colors` function will work even if the underlying implementation changes.

⚠ Removing the **Correct** answer with auto-grading for this question

1 +2.0
Correct

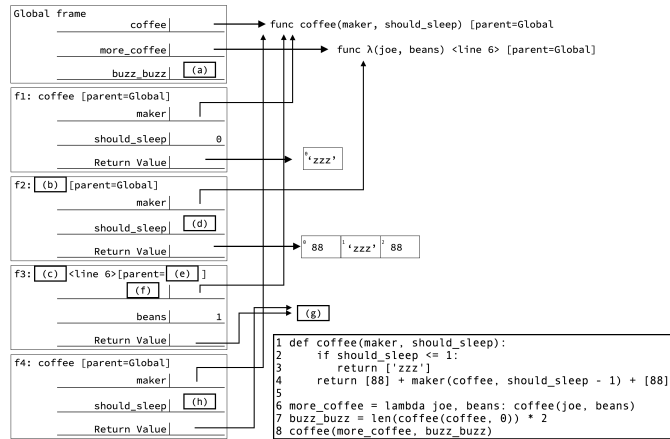
2 +0.0
Incorrect

+ Add Rubric Item

Q3 Coffee, Coffee, ☕

19 points

Fill in the blanks to complete the environment diagram. All the code used is in the box to the right, and the code runs to completion with no errors.



Q3.1

3 points

What is the value of **box (a)**?

- 0
- 1
- 2
- ['zzz']

⚠ Removing the **Correct** answer with auto-grading for this question

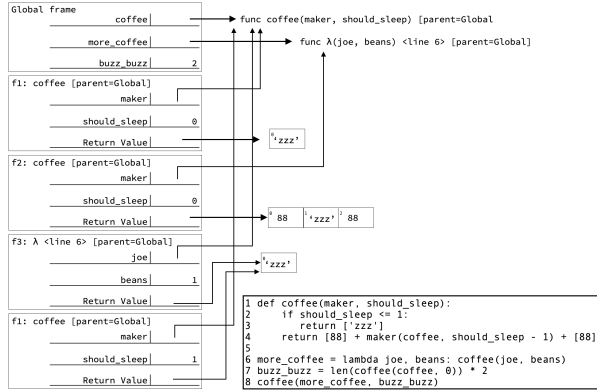
1 +3.0
Correct

2 +0.0
Incorrect

+ Add Rubric Item

Explanation

2 The full diagram is this:



Q3.2

2 points

What is the value of **box (b)**?

λ

coffee

more_coffee

maker

Explanation

coffee

⚠ Removing the **Correct** answer with auto-grading for this question

1 +2.0

Correct

2 +0.0

Incorrect

+ Add Rubric Item

Q3.3

2 points

What is the value of **box (c)**?

⚠ Removing the **Correct** answer with auto-grading for this question

λ

coffee

more_coffee

maker

Explanation

λ

1 +2.0
Correct

2 +0.0
Incorrect

[+ Add Rubric Item](#)

Q3.4

2 points

What is the value of **box (d)**?

0

1

2

['zzz']

Explanation

2

⚠ Removing the **Correct** attribute with auto-grading for this item

1 +2.0
Correct

2 +0.0
Incorrect

[+ Add Rubric Item](#)

Q3.5

2 points

What is the value of **box (e)**?

⚠ Removing the **Correct** attribute with auto-grading for this item

1 +2.0
Correct

Global

coffee

f1

f2

2 +0.0

Incorrect

+ Add Rubric Item

Explanation

Global

Q3.6

2 points

What is the value of **box (f)**?

joe

coffee

maker

⚠ Removing the **Correct** at with auto-grading for this

1 +2.0

Correct

Explanation

joe

2 +0.0

Incorrect

+ Add Rubric Item

Q3.7

2 points

What is the value of **box (g)**?

⚠ Removing the **Correct** at with auto-grading for this

1 +2.0

Correct

0

[]

['zzz']

[88, 88]

Explanation

['zzz']

2 +0.0

Incorrect

+ Add Rubric Item

Q3.8

2 points

What is the value of **box (h)**?

0

1

2

['zzz']

Explanation

1

⚠ Removing the **Correct** answer with auto-grading for this question

1 +2.0

Correct

2 +0.0

Incorrect

+ Add Rubric Item

Q3.9

2 points

What is the final result of line 8, `coffee(more_coffee, buzz_buzz)`?

⚠ Removing the **Correct** answer with auto-grading for this question

1 +2.0

Correct

['zzz']

[88, 88]

[88, 'zzz', 88]

[88, 88, 'zzz', 88, 88]

2 +0.0

Incorrect

+ Add Rubric Item

Explanation

[88, 'zzz', 88]

Q4

10 points

Which of the following HOFs should we use to solve this problem? Assume we can't use `sort()`, or `len()` or any other methods that work on a list directly, but any valid map function, filter function, etc. could be used.

If we say "reduce first" then map, we mean that the reduce call happens before we do a map. e.g.

```
value = reduce(some_function, sequence)
result = map(..., ...)
```


We can use `value` as a part of our mapper OR as our sequence.

Q4.1

2 points

Problem:

Input: A list of words

 Removing the **Correct** answer with auto-grading for this question.

Output: A sequence with words whose first letter is a vowel.

map

filter

reduce

map first then reduce

reduce first then map

reduce first then filter

filter first then map

filter first then reduce

Not Possible

1 +2.0

Correct

2 +0.0

Incorrect

+ Add Rubric Item

Explanation

This is a straightforward filter, with
 filter(lambda w: w[0] in 'aeiou', words)

Q4.2

2 points

Problem:

Input: A list of numbers

Output: The product of all numbers which are prime.

⚠ Removing the **Correct** answer with auto-grading for this item

1 +2.0

Correct

2 +0.0

Incorrect

+ Add Rubric Item

map
 filter
 reduce
 map first then reduce
 reduce first then map
 reduce first then filter
 filter first then map
 filter first then reduce
 Not Possible

Explanation

We can first filter on `is_prime`, then reduce with a `mul` function, or `lambda x, y: x*y` So `reduce(mul, filter(is_prime, numbers))`


Q4.3

2 points

Problem:

Input: A list of numbers

Output: All the elements in a list whose value is within 5 of the largest value in the list. (i.e. if the largest item were 10, find all the items > 5 . You do not know the largest item.)

 Removing the **Correct** answer with auto-grading for this question

1 +2.0
Correct

2 +0.0
Incorrect

 Add Rubric Item

map
 filter
 reduce
 map first then reduce
 reduce first then map
 reduce first then filter
 filter first then map
 filter first then reduce
 Not Possible

Explanation

This one is tricky, because the reduce comes as a part of the *function* in the filter expression.

```
filter(lambda x: x > reduce(max, numbers) - 5, numbers).
```

Admittedly, the reduce expression inside a filter is not super efficient, but you could write the following:

```
max_value = reduce(max, numbers)
```

```
filter(lambda x: x > max_value - 5, numbers)
```

max built in, but you could write a lambda as well.


Q4.4

2 points

Problem:

Input: A list of words, e.g. ['once', 'upon', 'a', 'time']

Output: A list of all the letters used, without duplicates, e.g. ['e', 'm', 't', 'n', 'c', 'i', 'p', 'a', 'o', 'u']

 Removing the **Correct** answer with auto-grading for this question.

1 +2.0
Correct

2 +0.0
Incorrect

 Add Rubric Item

map

filter

reduce

map first then reduce

reduce first then map

reduce first then filter

filter first then map

filter first then reduce

Not Possible

Explanation

This is easiest to envision as
`reduce(append_unique, map(word_to_list, words))`,
but technically you can have a more complicated
reducer function which accepts a word or a list of
letters, and outputs a list of letters.

Q4.5


2 points

Problem: Word Count

Input: A list of words, e.g. ['once', 'upon', 'a', 'time']

Output: A dictionary of the number of times each
word is used, e.g.

`{'once': 1, 'upon': 1, 'a': 1, 'time': 1}`

 Removing the **Correct** answer with auto-grading for this item

1 +2.0
Correct

2 +0.0
Incorrect

 Add Rubric Item

map
 filter
 reduce
 map first then reduce
 reduce first then map
 reduce first then filter
 filter first then map
 filter first then reduce
 Not Possible

Explanation

This one requires a little bit of creativity, but totally works. We can `map()` each word to a 1 item dictionary, then `reduce` with a function that combines two dictionaries. e.g.

```
reduce(dict_combiner, map(lambda w: {w: 1}, words)).
```

Our combiner function wouldn't be terribly complicated, but would take a few lines.

Q5

17 points

Consider the following `Tree` class:

```
class Tree:
    name = 'Tree'

    def __init__(self, value, branches=()):
        self.value = value
        for branch in branches:
            assert isinstance(branch, Tree), "Branches must be trees"
        self.branches = list(branches)

    def __repr__(self):
        """e.g. Tree(1, [Tree(2)])"""
```

```

if self.branches:
    branches_str = ',' + repr(self.branches)
else:
    branches_str = ''
return f'{self.name}({self.value}{branches_str})'

def is_leaf(self):
    return not self.branches

def add_branch(self, tree):
    assert isinstance(tree, Tree)
    self.branches.append(tree)

```

Q5.1

3 points


Let's say we wanted to make a new class `LinkTree` that enforced that we have only *one* branch. Which of the following class definitions should we use?

This class should inherit from a `Tree` and have all the methods of a `Tree`, but ensure that there's ever only 1 branch. Invalid `LinkTree`s should raise a `LinkTreeError`.

```
class LinkTreeError(Exception):
    pass
```

```
class LinkTree(__(a)__):
    name = 'LinkTree'
    def __init__(self, value, branches=()):
        self.value = value
        for branch in branches:
            assert isinstance(branch, __(b)__)
        if len(branches) > 1:
            __(c)__ LinkTreeError('LinkTree can only have one branch')
        self.branches = list(branches)

def add_branch(self, item):
```

 Removing the **Correct** answer with auto-grading for this question.

1 +3.0
Correct

2 +0.0
Incorrect

 Add Rubric Item

```

__(some changes omitted)__
assert isinstance(item, LinkTree)
self.branches.append(item)

```

What should go in the space __(a)__?

If nothing should go in that space, write 'nothing'.

Tree

Explanation

Tree , the class we are inheriting from.

Q5.2

2 points


What should go in the space __(b)__?

If nothing should go in that space, write 'nothing'.

LinkTree

Explanation

LinkTree , since if this were a linked list tree, we'd want all items to be the same type.

 Removing the **Correct** answer with auto-grading for this question.

1 +2.0
Correct

2 +0.0
Incorrect

[+ Add Rubric Item](#)


Q5.3

2 points

What should go in the space __(c)__?

If nothing should go in that space, write 'nothing'.

raise

 Removing the **Correct** answer with auto-grading for this question.

1 +2.0

Explanation

raise is the keyword for explicitly causing an error.

Correct

2 +0.0

Incorrect

+ Add Rubric Item

Q5.4

3 points

We want to adapt reduce to work on Trees. We're going to add a new method to our class.

```

from functools import reduce
def reduce(self, func):
    """Reduce the values of a tree to a single value.
    >>> tree = Tree(1, [Tree(2), Tree(3)])
    >>> tree.reduce(lambda x, y: x + y)
    6
    >>> tree = Tree(1, [Tree(2, [Tree(3)]), Tree(4)])
    >>> tree.reduce(lambda x, y: x * y)
    24
    """
    if ____(a)__.is_leaf():
        return self.value
    else:
        results = [__(b)___] + [__(c)___ for branch in self.branches]
        return reduce(__(d)___, results)

```

⚠ Removing the **Correct** answer with auto-grading for this question is not recommended.

1 +3.0

Correct

2 +0.0

Incorrect

+ Add Rubric Item

What should go in the space `__(a)___` ?

If nothing should go in that space, write 'nothing'.

self

Explanation

self , since this method exists on each item of a Tree .

Q5.5

2 points

What should go in the space ___(b)___ ?

- self.value
- value
- self
- self.branches

Explanation

self.value

⚠ Removing the **Correct** answer with auto-grading for this item

1 +2.0
Correct

2 +0.0
Incorrect

+ Add Rubric Item

Q5.6

3 points

What should go in the space ___(c)___ ?

- self.reduce(func, branch)
- self.reduce(func)
- branch.reduce(func)
- self.reduce(branch)

⚠ Removing the **Correct** answer with auto-grading for this item

1 +3.0
Correct

2 +0.0
Incorrect

+ Add Rubric Item

Explanation

branch.reduce(func) - the reduce method exists on each Tree instance so we call branch.reduce .

Q5.7

2 points

What should go in the space ___(d)___ ?

self.value

self.func

self

func

Explanation

The final step is to pass func to the normal reduce method.

⚠ Removing the **Correct** answer with auto-grading for this question

1 +2.0
Correct

2 +0.0
Incorrect

+ Add Rubric Item

Q6 Efficiency

6 points

For each of the following questions, select the time complexity of the code provided. Assume that the input is nonnegative.

Q6.1

2 points

```
def five_n_plus_two(n):
    if n == 0:
        return 2
    else:
        return 5 + five_n_plus_two(n - 1)
```

 $O(1)$ $O(n)$ $O(n^2)$ $O(2^n)$

Explanation

This is a simple linear function. Since we make n calls to compute the sum.

Q6.2

2 points

```
def five_n_plus_two(n):
    s = 2
    for _ in range(5):
        s += n
    return s
```

 $O(1)$ $O(n)$ $O(n^2)$ $O(2^n)$

⚠ Removing the **Correct** answer with auto-grading for this question.

1 +2.0
Correct

2 +0.0
Incorrect

+ Add Rubric Item

⚠ Removing the **Correct** answer with auto-grading for this question.

1 +2.0
Correct

2 +0.0
Incorrect

+ Add Rubric Item

Explanation

This is "constant" or $O(1)$ because no matter how large n is, the loop executes 5 iterations.

Q6.3


2 points

```
def add_numbers(n):
    s = 0
    for i in range(1, n):
        for z in range(1, i):
            s += z
    return s
```

 $O(1)$ $O(n)$ $O(n^2)$ $O(2^n)$

Explanation

This one is a little less obvious, but we still have n^2 . We have a n iterations of our outer loop, then our inner loop executes once, then twice, then 3, 4, 5, and so on times up until the last loop where it executes n times. So in reality, it's not *quite* $n * n$ iterations, but that's an *upper bound* for the run time.

 Removing the **Correct** answer with auto-grading for this question

1 +2.0
Correct

2 +0.0
Incorrect

 Add Rubric Item

Q7 Generating Generators Generates Generators

4 points

We're going to write a function that turns another function into a generator that can be called `max_times`.

```
def generator_generator(func, max_times):
    """
    >>> numbers = generator_generator(lambda x: x + 1, 4)
    >>> list(numbers)
    [1, 2, 3, 4]
    >>> sum(generator_generator(lambda x: x + 1, 10))
    55
    """
```

Put the following lines of code in the correct order:

- (a) `while counter < max_times:`
- (b) `yield func(counter)`
- (c) `counter += 1`
- (d) `counter = 0`

Indentation does not matter.

Solution:

```
def generator_generator(func, max_times):
    """
    >>> numbers = generator_generator(lambda x: x + 1, 4)
    >>> list(numbers)
    [1, 2, 3, 4]
    >>> sum(generator_generator(lambda x: x + 1, 10))
    55
    """
    (d) counter = 0
    (a) while counter < max_times:
        (b) yield func(counter)
        (c) counter += 1
```

Q7.1

1 point


Line 1:

A

B


C

D

 Removing the **Correct** answer with auto-grading for this question.

1 +1.0
Correct

2 +0.0
Incorrect

 Add Rubric Item

Q7.2

1 point


Line 2:

A

B


C

D

 Removing the **Correct** answer with auto-grading for this question.

1 +1.0
Correct

2 +0.0
Incorrect

 Add Rubric Item

Q7.3

1 point

Line 3:

- A
- B
- C
- D

⚠ Removing the **Correct** answer with auto-grading for this item

1 +1.0
Correct

2 +0.0
Incorrect

+ Add Rubric Item

Q7.4

1 point

Line 4:

- A
- B
- C
- D

⚠ Removing the **Correct** answer with auto-grading for this item

1 +1.0
Correct

2 +0.0
Incorrect

+ Add Rubric Item

Q8 SQL

8 points

Q8.1

2 points

The programming language SQL best models which paradigm?

Imperative Programming

Array-Based Programming

Functional Programming

Object-Oriented Programming

Declarative Programming

⚠ Removing the **Correct** answer with auto-grading for this question

1 +2.0
Correct

2 +0.0
Incorrect

Explanation

SQL is a declarative programming language.

+ Add Rubric Item

Q8.2

2 points

Put the SQL keywords in the right order they must be used in order for a query to work. Not all of these parts of a query are necessary, but if they are all there, there is only one correct order.

ORDER

FROM

GROUP BY

LIMIT

JOIN

WHERE

SELECT

⚠ Removing the **Correct** answer with auto-grading for this question

1 +2.0
Correct

2 +0.0
Incorrect

+ Add Rubric Item

SELECT FROM GROUP BY LIMIT JOIN
 WHERE ORDER

SELECT JOIN FROM WHERE GROUP BY
 LIMIT ORDER

SELECT FROM ORDER WHERE GROUP BY
 JOIN LIMIT

SELECT FROM JOIN WHERE GROUP BY
 ORDER LIMIT

SELECT JOIN FROM GROUP BY WHERE
 ORDER LIMIT

Explanation

SELECT FROM JOIN WHERE GROUP BY
 ORDER LIMIT


Q8.3

2 points

For the following two questions consider these two tables:

drinks

id	name	syrup_id
1	pumpkin spice latte	1
2	strawberry latte	4
3	mocha	2
4	peppermint hot chocolate	3
5	vanilla latte	5

 Removing the **Correct** answer with auto-grading for this question

1 +2.0
Correct

2 +0.0
Incorrect



id	name	syrup_id
6	strawberry frappuccino	4

syrups

id	name	price
1	pumpkin spice	0.25
2	chocolate	0.50
3	pettermint	0.75
4	strawberry	1.00
5	vanilla	1.25

How many rows are in the cartesian product of the two tables drinks and syrups ?

- 5
- 6
- 11
- 30
- 25
- 36

Explanation

30. The cartesian production is the total number of rows in each table multiplied together.

Q8.4

2 points

How many rows would the returned table have if we JOIN the tables with the condition drinks.syrup_id = syrups.id ?

5

6

11

30

25

36

Explanation

6. Each row in the drinks table (6 words) would have exactly 1 row in the syrup table which matches the syrup_id.

⚠ Removing the **Correct** answer with auto-grading for this question

1 +2.0
Correct

2 +0.0
Incorrect

+ Add Rubric Item

Q9

8 points

Please fill out the post-exam proctoring form.

<https://go.c88c.org/proctoring-form>

You'll get points for this question as long as you fill out the proctoring form.

Optionally, tell us how you're feeling!

1 +8.0
Submitted the proctoring form

2 +0.0
No proctoring form submitted

+ Add Rubric Item

